Voltage and Current Based Fault Simulation for Interconnect Open Defects

Haluk Konuk

Abstract

This paper describes a highly accurate and efficient fault simulator for interconnect opens in combinational or full-scan digital CMOS circuits. The analog behavior of the wires with interconnect opens are modeled very efficiently in the vicinity of the defect in order to predict what logic levels the fanout gates will interpret, and whether a sufficient IDDQ current will be flowing inside the fanout gates. The fault simulation method is based on characterizing the standard cell library with SPICE; using transistor charge equations for the site of the open; using logic simulation for the rest of the circuit; taking four different factors that can affect the voltage of an open into account; and considering the potential oscillation and sequential behavior of interconnect opens. The tool can simulate test vectors for both voltage and current measurements. Simulation results of ISCAS85 layouts using stuck-at and IDDQ test sets are presented.

1 Introduction

Breaks are a common type of defects that occur during an IC manufacturing process [1]. Breaks in a digital CMOS circuit fall into different categories depending on their location. A break can occur inside a CMOS cell affecting transistor drain and source connections [2, 3, 4, 5], disconnect a single transistor gate from its driver [6, 7], or disconnect a set of logic-gate inputs from their drivers; thus causing these inputs to electrically float. In order for a break to disconnect a set of logic-gate inputs from their drivers, the break must occur in the interconnect wiring. In today's CMOS ICs with up to five metal layers, interconnect wiring is probably the most likely place for a break to occur. This is well supported by the critical area analysis by Xue, *et al.* [8]. Also, vias are especially susceptible to breaks [9], and the number of vias exceeds the number of transistors in some microprocessor designs [10].

We call the fault created by a break in the interconnect wiring an **interconnect open**. In order to make our terminology more specific, a break refers to a discontinuity caused by a manufacturing defect in the physical layout of a design, and an open refers to the corresponding discontinuity defect in the electrical circuit of that design. In this paper we describe a fault simulation algorithm for interconnect opens

in combinational or full-scan circuits that take into account all known factors affecting the voltage of an electrically floating wire created by an interconnect open. The end result is the prediction of the analog behavior of the floating wire such that the logic level that is seen by each gate driven by the floating wire is determined. Moreover, for a given vector whether sufficient IDDQ current will flow inside the gates driven by the floating wire is also determined.

This paper is an extended version of the work described at ICCAD-97 [11]. To the best of our knowledge, this is the first fault simulator reported for interconnect opens; therefore, comparison to prior work is not applicable. Note that the fault simulator reported in [2] targets **network breaks**, which affect only the transistor drain and source connections inside a CMOS cell. By definition, a set formed by network breaks is disjoint from a set formed by interconnect opens. Therefore, the two fault simulators have disjoint fault spaces. In the following section, we discuss the factors affecting the voltage of a floating wire. Section 3 describes the pre-processing of the standard-cell library, whose results are used during fault simulation. Section 4 describes the fault simulation algorithm, and finally Section 5 presents the results of our experiments with ISCAS85 circuits using stuck-at and IDDQ test sets.

2 The Determining Factors for the Floating Wire Voltage

Knowing the factors that determine the voltage of a floating wire is a basic requirement for building a fault simulator for interconnect opens. We consider the factors discussed in the following subsections as the most important ones, and we assume that any other factor, such as conduction through the silicon-dioxide between metal lines is negligible [12]. We do not cover here due to lack of space how the effect of charge collector diodes can be handled, which can be found in Chapter 4 in [13].

2.1 Capacitances between the floating wire and its neighboring wires

We illustrate the types of wiring capacitances that contribute to the voltage of a floating wire with the help of Figure 1. In this figure an example of an interconnect open is shown. C_5 denotes the total wiring capacitance from floating wire FW to the n-wells and to the V_{DD} supply wires. C_6 is the total wiring capacitance from FW to the substrate and to the GND supply wires. C_3 and C_4 denote the wire-to-wire capacitances from FW to neighboring signal wires, which are created by metal tracks running next to, over, or under FW. The sizes of C_3, C_4, C_5, and C_6 together with the voltages on signal lines $S3$ and $S4$ contribute to the determination of the voltage on FW. The meanings of V_{surf} and C_{surf} are discussed later.

One needs to know the exact location of an open in the interconnect in order to obtain the list and sizes of wiring capacitances to the corresponding floating wire. If an open is going to occur on a piece of

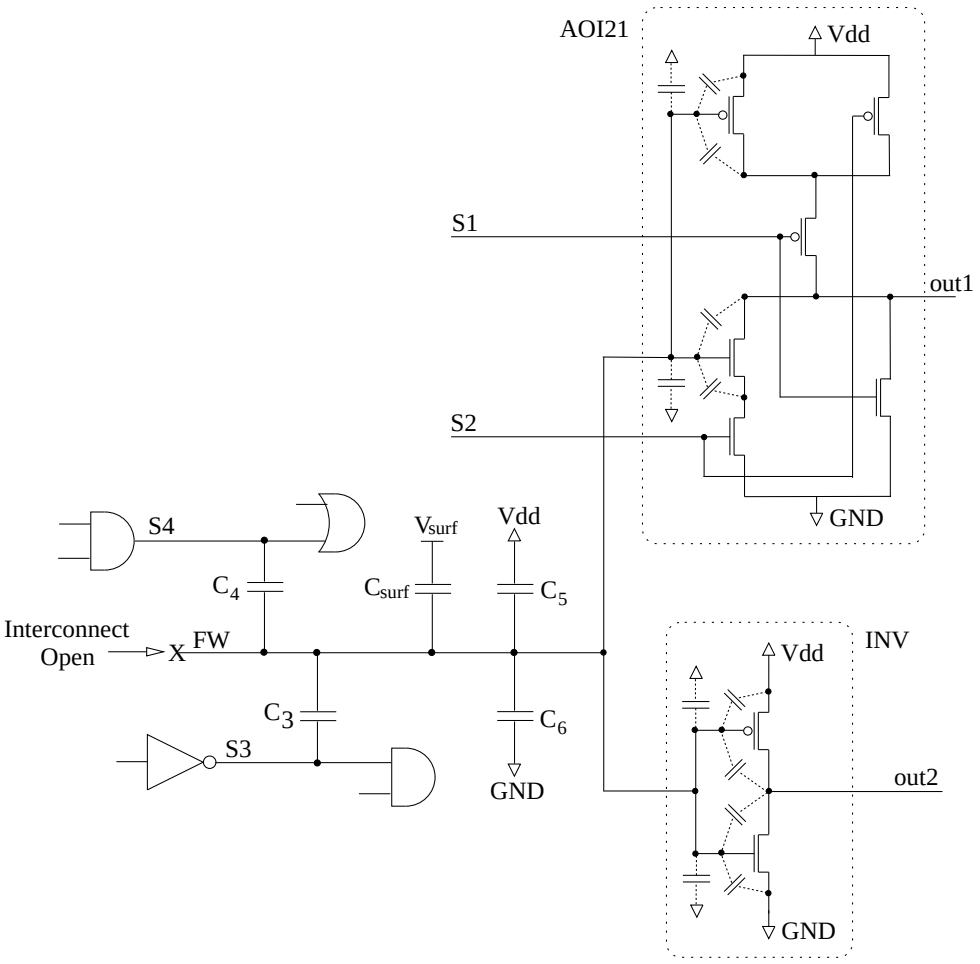

Figure 1: A floating wire created by an interconnect open.

straight metal track, we are not aware of any way of predicting where the open defect will land on this metal track. Besides, contacts are much more susceptible to opens than metal tracks are, according to the defect distribution statistics by Feltham and Maly [9]. Also, the increasing number of metal layers in IC processes tends to increase the number of vias per metal layer. In this work, each via, that produces a floating wire when broken, is considered to be a potential interconnect open defect site. The fault list for our simulator is produced by considering each such via.

Accuracy in wiring capacitance estimation is another important issue. There are two sources of inaccuracy for wiring capacitances: (1) computation of the capacitances from the layout, which is called capacitance extraction, and (2) manufacturing process variations.

In our experiments we used *Magic* for capacitance extraction, which can perform a 2-D (two-dimensional) extraction, based on the area and the perimeter of the overlap between two conducting surfaces. As an example for the inaccuracy of 2-D extraction, consider the capacitance between two parallel metal-2 wires A and

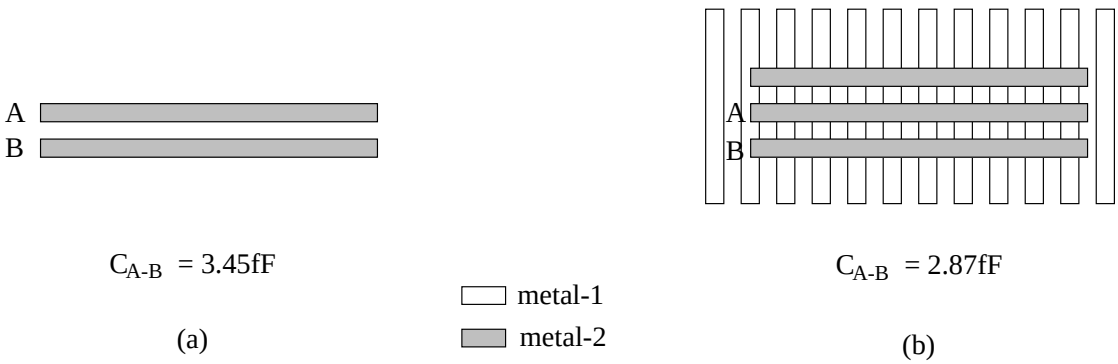

Figure 2: Wire-to-wire capacitance variation due to surrounding topology.

B in Figure 2. Both wires are 0.9μ by 60.0μ separated by 0.9μ. Using a 3-D (three-dimensional) capacitance extraction program, called SPACE3D [14], and wire height and silicon-dioxide thickness parameters for an HP 0.6μ process, we obtained 3.45fF when there are no other wires in the surroundings, as shown in part (a) of Figure 2. However, when there are many other metal-1 and metal-2 wires in the surroundings interfering with the electric field lines between A and B as shown in part (b), the capacitance goes down to 2.87fF. *Magic* extracts 2.88fF for both cases, while the actual capacitance in part (a) is about 20% larger than the one in part (b). Our additional experiments with a metal-2 wire crossing a metal-1 wire at a right angle showed about 70% variation for the cross-over capacitance depending on the surrounding topology.

Variations in the thickness of the inter-metal dielectric and the thickness of the metal will cause variations in the actual wiring capacitances. Even though manufacturing techniques, such as CMP (chemical mechanical polishing), reduce these variations, they still need to be taken into account. Furthermore, the line width variation will cause significant capacitance variation since with high aspect ratios, the bulk of the capacitance will be to adjacent lines.

One way to deal with the variations in the wiring capacitances is to assume that the actual capacitance is within +/- x% of the extracted capacitance, where the value of x is determined by the type of extraction tool used and the manufacturing process. This way, our fault simulator uses a range for each wiring capacitance in order to compute a voltage range for a given open.

2.2 Transistor capacitances to the floating wire

These capacitances are in the cells driven by *FW* in Figure 1. They are gate/drain, gate/source, and gate/bulk capacitances, which are connected to transistor terminals with dotted lines to emphasize that they are not additionally inserted, but they are part of any CMOS transistor. The bulk terminals of p- and n-channel transistors are connected to V_{DD} and GND, respectively. Values of transistor capacitances

significantly vary depending on the transistor terminal voltages. For this reason, we use transistor gate charge equations expressed as functions of transistor terminal voltages [15, 2] and transistor geometry, rather than using fixed worst case capacitance values to compute the charge stored on transistor gates.

Even though transistor geometries can vary due to process variations causing some variation in transistor capacitances, theses capacitances are much less affected by surrounding structures than wiring capacitances are, because gate-oxide thickness is on the order of $100A^{\circ}(= 0.01\mu)$ while metal wires are separated by about 1μ or so in a 0.6μ technology. In this paper, we do not use a value range for a transistor capacitance as we do for a wiring capacitance.

2.3 Trapped charge deposited on the floating wire

Experiments by Johnson [16] and Konuk and Ferguson [12] showed that the trapped charge deposited during fabrication can build up a voltage from -4.0V to 2.3V on floating gates with poly extensions, and -1.0V to 1.0V on floating gates connected to metal wires. We are not aware of any technique to predict the amount of trapped charge on a particular interconnect open. Therefore, our fault simulator makes no assumptions for the amount of the actual trapped charge. However, if the trapped charge is known to be within a range for any interconnect open, then this range can be given to our fault simulator as an input.

2.4 The RC interconnect behavior of the die surface

Konuk and Ferguson [12] reported an experimental observation that the die surface acted as an RC interconnect, capacitively coupling the floating wire to almost all other signals in a chip. The die surface resistance for this phenomenon to occur is in the tera-Ohms range. When coupled with wiring capacitances in the femto-Farads range in a chip, a time constant of one second or less is produced. Therefore, C_{surf} in Figure 1 might need to be taken into account when determining the voltage of FW by using a worst case value for the surface voltage V_{surf}. C_{surf} in Figure 1 denotes the capacitance between the floating wire and the die surface, and V_{surf} denotes the voltage at the portion of the die surface right above the floating wire.

3 Processing the Standard-Cell Library

Before performing fault simulation on a particular standard-cell based design, the standard-cell library needs to be processed. We assume that each cell in the library is either a basic cell or is composed of basic cells, where we define a **basic cell** as a network of p-channel transistors and a complementary network of n-channel transistors, where each cell input drives the gates of one p- and one n-channel transistor, such as the AOI21 and INV cells in Figure 1. All the MCNC [17] cells used in the ISCAS85 circuit layouts satisfy this condition.

Our fault simulator will require some modification in order to support non-basic cells.

During the library preprocessing, we first determine the $\boldsymbol{L0\_th}$ and $\boldsymbol{L1\_th}$ values, which denote the maximum voltage that is still logic-0 and the minimum voltage that is still logic-1 for the cell library, respectively [2], which we computed as 1.05V and 1.90V for the MCNC library using HP 0.6μ HSPICE [18] level-13 parameters (the BSIM model) from MOSIS [19], and setting V_{DD} to 3.3V.

We define a **composite input** or a **c-input** for a cell as either a single cell input or multiple input ports of the same cell tied together. $\boldsymbol{V_{L0,g,ci}}$ denotes the voltage on the c-input ci of cell g such that the output of g is at L1_th, and is sensitized to ci. For instance; 1.302V on the a input of the 2-input NAND gate in the MCNC library creates 1.90V (the L1_th voltage) on the NAND gate's output when the b input is 3.3V. Therefore, V_{L0} is 1.302V for the c-input consisting of only the a input of the NAND gate. $\boldsymbol{Q_{L0,g,ci}}$ denotes the total electrical charge on the transistor gates that are driven by ci, when the voltage on ci is $V_{L0,g,ci}$. $\boldsymbol{V_{L1,g,ci}}$ and $\boldsymbol{Q_{L1,g,ci}}$ are similarly defined.

Note that the voltage levels on other inputs of g can affect the threshold values $V_{L0,g,ci}$ and $V_{L1,g,ci}$ in case of complex cells. For instance; the output of the OAI22 cell shown in Figure 5 will be sensitized to its $b2$ input when $b1 = 0$, $a1 = 0$, and $a2 = 1$. However, the output will still be sensitized to $b2$ even when $a1$ changes from 0 to 1. The difference is that in the first case there is only one current path, which goes through the nMOS transistor driven by $a2$. In the second case, when $a1$ becomes 1, there are two current paths going through the bottom two nMOS transistors. The V_{L0} and V_{L1} values will slightly change depending on which sensitization case is used. In our processing of the MCNC standard cell library, we used the sensitization conditions for each complex cell such that the V_{L0} values are minimized and the V_{L1} values are maximized.

For IDDQ testing [20] the traditional approach has been to determine a threshold current $\boldsymbol{I_{DDQ,th}}$ such that a quiescent power supply current larger than $I_{DDQ,th}$ indicates a defective chip. As the number of transistors on a chip is increasing together with the increasing leakage current per transistor and its variation from die to die, it is becoming very difficult to find a single IDDQ threshold that can differentiate a defective chip from a defect-free one. One attempt to overcome this problem is the current signatures method [21]. In this method the difference in the IDDQ currents between different IDDQ vectors is important rather than the magnitudes of the currents only. If a large step in IDDQ is observed with some of the vectors, that will be the indication of a defective chip. In the rest of this paper, $I_{DDQ,th}$ can be interpreted as the minimum amount of additional IDDQ current caused by an interconnect open such that this open will be detected by the particular IDDQ technique used, that is, either the single threshold method or the current signatures method.

Let $\boldsymbol{V_{I_{DDQ}0,g,ci}}$ denote the logic-0 voltage on c-input ci such that $I_{DDQ,th}$ flows through cell g. Let $\boldsymbol{Q_{I_{DDQ}0,g,ci}}$ denote the total electrical charge on the transistor gates that are driven by ci, when the voltage

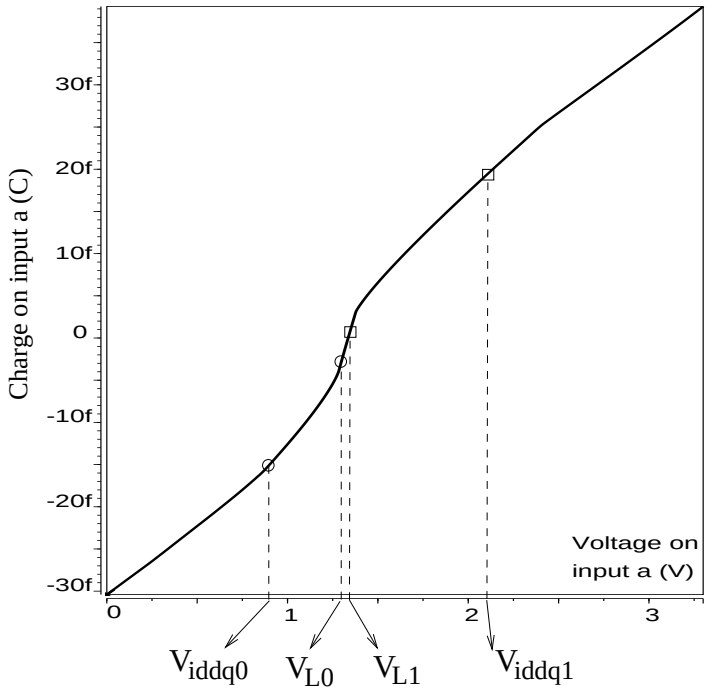

Figure 3: Charge versus voltage plot for the a input of a NAND gate.

on ci is $V_{I_{DDQ}0,g,ci}$, and $I_{DDQ,th}$ flows through g. $\boldsymbol{V_{I_{DDQ}1,g,ci}}$ and $\boldsymbol{Q_{I_{DDQ}1,g,ci}}$ are similarly defined.

For every cell g and c-input ci in the library, $V_{L0,g,ci}$, $V_{L1,g,ci}$, $V_{I_{DDQ}0,g,ci}$, $V_{I_{DDQ}1,g,ci}$, $Q_{L0,g,ci}$, $Q_{L1,g,ci}$, $Q_{I_{DDQ}0,g,ci}$, and $Q_{I_{DDQ}1,g,ci}$ are computed and recorded using HSPICE. In addition, the slope and the y-intercept values for the straight line defined by points ($V_{I_{DDQ}0,g,ci}$, $Q_{I_{DDQ}0,g,ci}$) and ($V_{L0,g,ci}$, $Q_{L0,g,ci}$) are recorded to be used for charge interpolation in our fault simulation algorithm. Similarly, the slope and the y-intercept for the straight line defined by points ($V_{L1,g,ci}$, $Q_{L1,g,ci}$) and ($V_{I_{DDQ}1,g,ci}$, $Q_{I_{DDQ}1,g,ci}$) are recorded. For example; consider the HSPICE charge-voltage plot in Figure 3 for the a input of the 2-input NAND gate in the MCNC library using HP 0.6μ process parameters. The $V_{I_{DDQ}0}$, V_{L0}, V_{L1}, and $V_{I_{DDQ}1}$ points are marked using $I_{DDQ,th} = 50\mu A$.

4 Fault Simulation Algorithm

Figure 4 shows the very top level structure of our algorithm. After all the vectors and opens are processed, an interconnect open is marked detected if the detection range for the trapped charge spans from $-\infty$ to $+\infty$ if the trapped charge cannot be bounded.

Let $V_{Q_{trapped},FW}$ denote the voltage created by the trapped charge on interconnect open FW when the chip is unpowered, that is, when all other circuit nodes are at GND voltage. Our program lets the user

```
FOREACH 32-vector DO
    FOREACH interconnect open DO
        Perform PPSFP (parallel pattern single fault propagation) [22] using the 32 vectors
        after flipping the fault-free logic values on the floating wire created by the interconnect open.
        FOR vector 1 THROUGH 32 DO
            IF vector can detect the open THEN
                Compute the range of trapped charge for this vector to detect the open, and
                add this range to the detection ranges computed from previous vectors.
ENDFOR; ENDFOR; ENDFOR
```

Figure 4: Top level structure of the fault simulation algorithm

specify a voltage range bounding the trapped charge, such that $V_{Q_{trapped},FW}$ for any FW is guaranteed to be within this range. In order to be able to give this range, the user will need to have some knowledge about the manufacturing process. If specified, in order to decide whether an open is detected or not this range will be compared against the trapped charge ranges computed by our program that are necessary for detection.

Detection of a defect can be accomplished by either voltage sensing or current sensing (IDDQ testing) or both. If voltage sensing is used when a test set is applied, then an interconnect open will be detected if it behaves like a stuck-at fault for at least one vector of the test set, which covers this stuck-at fault. If current sensing is used, then an interconnect open will be detected by a vector if at least one of the cells driven by the floating wire draws an IDDQ current larger than $I_{DDQ,th}$. The following four subsections describe how the detection range for trapped charge is computed in cases of voltage sensing, current sensing, and both.

4.1 Voltage sensing (stuck-at detection)

For a given vector and a given logic value on floating wire FW, let us first describe how we split a c-input of a cell driven by FW into sensitized and unsensitized parts. A **sensitized c-input** is formed by those input ports of a c-input, which drive those transistors through which an IDDQ current flows if the c-input voltage turns both p- and n-channel transistors on, while all other inputs of the cell are kept at V_{DD} or GND voltage. The remaining part of the c-input is called an **unsensitized c-input**. For a simple gate, such as a 2-input NAND gate, the c-input formed by tying its both input together is also a sensitized c-input, because if the voltage on this c-input can turn both p- and n-channel transistors on, then a static current will be flowing through all its transistors. However, the c-input formed by only the a input of this NAND gate would also

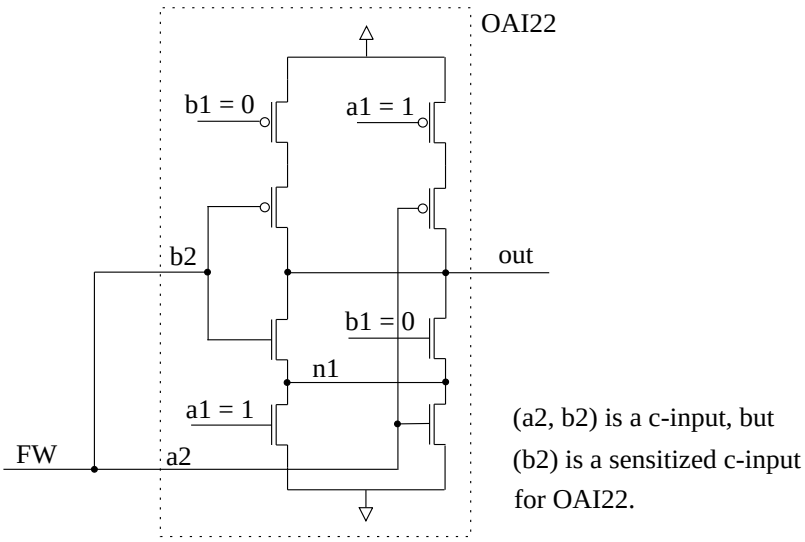

Figure 5: An example for a sensitized c-input

be a sensitized c-input only if the b input has the V_{DD} voltage on it, otherwise it would be an unsensitized c-input.

A more complicated example, where only a part of a c-input forms a sensitized c-input is illustrated by Figure 5. In this figure floating wire FW is driving a c-input that is formed by tieing the $a2$ and $b2$ ports of an OAI22 gate. We denote this c-input as ($a2$, $b2$). In this case with $a1 = 1$ and $b1 = 0$, ($b2$) is the sensitized, and ($a2$) is the unsensitized c-input portions, because the p- and n-channel transistors driven by $a2$ will not have a static current through them when both are turned on. Actually, the n-channel transistor driven by $a2$ will have a tiny amount of current, but this is negligible because of the parallel by-pass n-channel transistor that is fully turned on by $a1 = 1$.

If an applied vector detects a stuck-at-0 or a stuck-at-1 fault on the floating wire FW, then the first step to determine the trapped charge range for this vector to detect the open is to find the maximum FW voltage that is still logic-0 in case of stuck-at-0, or the minimum FW voltage that is still logic-1 in case of stuck-at-1. We assume the stuck-at-0 case for the rest of this subsection without loss of generality. We determine the maximum logic-0 voltage on FW for the applied vector vec, $\boldsymbol{V_{L0,FW,vec}}$, as the minimum logic-0 threshold voltage over all the sensitized c-inputs driven by FW. The following describes this more formally, where g is the cell the sensitized c-input ci belongs to. Recall that the $V_{L0,g,ci}$ values are pre-computed during the library processing step.

$V_{L0,FW,vec} = \infty$
FOREACH sensitized c-input ci driven by FW DO
    IF $V_{L0,g,ci} < V_{L0,FW,vec}$ THEN $V_{L0,FW,vec} = V_{L0,g,ci}$

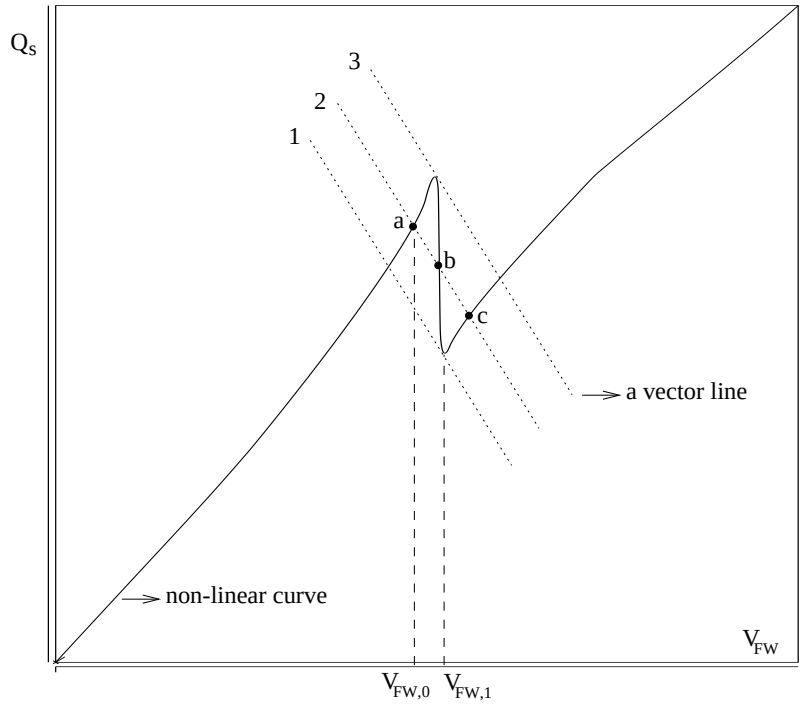

Figure 6: Sequential behavior caused by an interconnect open.

ENDFOR

Then, we compute the trapped charge on FW, that corresponds to a voltage of $V_{L0,FW,vec}$ on FW with *vec* applied to the circuit. Let us call this computed charge $\boldsymbol{Q_0}$. We defer the description of the charge computation to Section 4.4. Intuitively, if the actual trapped charge is smaller than Q_0, then the FW voltage will be smaller than $V_{L0,FW,vec}$, that is, the interconnect open will be detected as a stuck-at-0 fault by *vec* when the actual trapped charge is less than or equal to Q_0. We will next show that this is not always true.

Let us consider the case where there is a non-inverting combinational path from FW to $S3$ through $out1$ or $out2$ in Figure 1, and this path is sensitized by vector *vec*. Let $\boldsymbol{Q_s}$ denote the sum of the charges on transistor gates driven by FW and the charge on the FW plate of capacitor C3. The solid line in Figure 6 shows a typical HSPICE plot of Q_s versus V_{FW} [23, 13], which we call the **non-linear curve**. The reason for the steep drop in Q_s is as follows: When V_{FW} is equal to $V_{L0,FW,vec}$, a small increase in V_{FW} will push it over to logic-1 state, which in turn will cause a GND to V_{DD} transition on $S3$. This transition will push positive charge away from the FW plate of C3. The amount of this displaced charge is (V_{DD} * C3) if we ignore the small increase in V_{FW}. This is causing the drop in Figure 6. From the law of charge conservation:

$$Q_{trapped} = Q_s + C0 \cdot V_{FW} + C1 \cdot (V_{FW} - V_{DD}) + C_{surf} \cdot (V_{FW} - V_{surf}) \quad (1)$$

where C0 is the sum of capacitances from FW to the nodes at GND, and C1 is the sum of capacitances from FW to the nodes at V_{DD}. Also, the voltages on these nodes are not dependent on V_{FW}. Thus, if $S4$ is at logic-1, then C0 = C6, and C1 = C4 + C5. Equation 1 can be rewritten as

$$Q_s = A - B \cdot V_{FW} \ , \text{ where} \quad (2)$$

$$A = Q_{trapped} + C_{surf} \cdot V_{surf} + C1 \cdot V_{DD} \ , \ B = C0 + C1 + C_{surf}$$

For a given interconnect open, the value of B is fixed, that is, the slope of the straight line defined by Equation 2 is fixed. The value of A depends on C1 and $Q_{trapped}$ if we assume a fixed V_{surf} value. For a given vector, C1 is fixed; therefore, only $Q_{trapped}$ determines the offset of the straight line of Equation 2. This straight line is called **vector line** by Konuk [13], because the value of C1 can change with every vector applied to the circuit, which in turn changes the value of A, thus moving the straight line up or down. For a given open, a given vector, and a fixed V_{surf} value, vector lines 1, 2 and 3 in Figure 6 correspond to three different $Q_{trapped}$ values. The $Q_{trapped}$ for line 1 is smaller than the one for line 2, which is smaller than the one for line 3.

For a given V_{FW}, the $Q_{trapped}$ computation in Section 4.4 effectively finds the intersection between the non-linear curve and a vector line in Figure 6. For instance, for $V_{FW,0}$, which is a logic-0 state, the computed $Q_{trapped}$ corresponds to the one for line 2, called Q_0. Note that for $V_{FW,1}$, which is a logic-1 state, the non-linear curve intersects line 1, which is lower than line 2, thus has a smaller $Q_{trapped}$ value than line 2. This means that the floating wire can be at a logic-1 state with an actual trapped charge smaller than Q_0. Only when the actual trapped charge is smaller than the one for line 1, the open is guaranteed to be at logic-0 state. Therefore, the goal is to find the $Q_{trapped}$ value that corresponds to the vector line, which intersects the downward pointing elbow of the non-linear curve in Figure 6.

To achieve this goal, we recompute $Q_{trapped}$ again by using $V_{L0,FW,vec}$ with vec applied to the circuit, but this time with logic-1 on FW propagated to all circuit nodes that are sensitized to FW. We call this computed charge $\boldsymbol{Q_1}$. Conceptually, we could use $V_{FW,1}$ instead of $V_{L0,FW,vec}$, but it is not feasible to compute the value of $V_{FW,1}$, and using $V_{L0,FW,vec}$, which is expected to be close to $V_{FW,1}$, results in even a smaller trapped charge value.

Then, the maximum actual trapped charge with which vec is guaranteed to detect the open as a stuck-at-0 fault is $Q_{trapped,max,sa0} = min(Q_0, Q_1)$. In other words, the detection range for vector vec is $(-\infty, Q_{trapped,max,sa0})$. The fact that a vector line can intersect the non-linear curve in Figure 6 at three points,

as line 2 does, shows that an interconnect open can have two stable states (points a and c), one at logic-0 and the other at logic-1; and one meta-stable state (point b) as explained in [23] and [13]. Therefore, the logic state of the interconnect open for line 2 can be either 0 or 1 for a given vector depending on the state of this open for the previous vector applied to the circuit. We call this **sequential behavior**, because the logic state of the open not only depends on the present input vector but also on history. As described above, the $Q_{trapped,max,sa0}$ value found corresponds to the vector line that intersects the downward pointing elbow of the non-linear curve in Figure 6, which is line 1. If the actual trapped charge is below this value, then the corresponding vector line will also be below line 1, which will intersect the non-linear curve at only one point. This guarantees that the logic state of the open will be 0 irrespective of the past history; thus avoiding sequential behavior.

In addition to the sequential behavior, an interconnect open can also oscillate under certain conditions as explained in more detail in [23, 13]. Oscillation can occur due to capacitance from a signal S to the floating wire FW, such that there is an inverting combinational path from FW to S, and that path is enabled by the given vector. Consider the case when a 0-to-1 or a 1-to-0 transition on S causes the voltage of FW move such that the same logic transition happens on FW due to the mutual capacitance. This will in turn cause a transition on S because of the enabled inverting path; thus, oscillation.

In the computation of Q_0 above, logic values at all signal nodes are found by placing logic-0 on FW. Therefore, any node that has an enabled inverting path from FW to it, such as S, will have a logic-1 on it. A logic transition on such a node can only be a 1-to-0 transition, which will push the voltage on FW more towards GND because of the mutual capacitance, which is the opposite direction for an oscillation to occur. Therefore, the computation of Q_0 is such that oscillation will not occur. The same reasoning is valid for the computation of Q_1, which is described earlier in this section. Thus, the computation of $Q_{trapped,max,sa0}$ guarantees that oscillation will not occur if the actual trapped charge is below this value.

4.2 Current sensing (IDDQ detection)

First, we would like to demonstrate how a change in the logic value of the floating wire can change the set of sensitized c-inputs. When FW in Figure 7 is logic-1, the floating input of G3 is a sensitized c-input. However, it is an unsensitized c-input when FW is logic-0. The floating input of G1 is a sensitized c-input in both cases.

For IDDQ detection, we first check whether the given vector vec and a logic-0 on FW create any sensitized c-input driven by FW. If so, we determine the minimum voltage on FW, $\boldsymbol{V_{I_{DDQ}0,FW,vec}}$, such that an IDDQ that is larger than or equal to $I_{DDQ,th}$ still flows through a cell driven by FW. Note that $V_{I_{DDQ}0}$ is less than V_{L0}, and $V_{I_{DDQ}1}$ is greater than V_{L1} for a basic cell, as illustrated in Figure 3. So, we determine

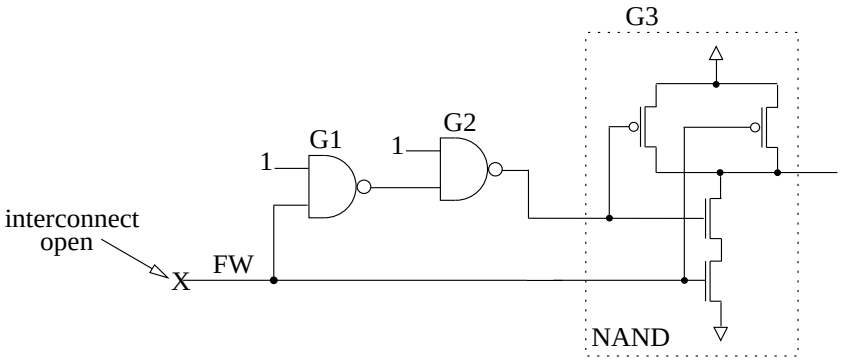

Figure 7: The effect of the floating wire's logic value on sensitized c-inputs

$V_{I_{DDQ}0,FW,vec}$ as the minimum logic-0 IDDQ threshold voltage over all the sensitized c-inputs driven by FW. The following describes this more formally, where g is the cell the sensitized c-input ci belongs to.

$V_{I_{DDQ}0,FW,vec} = \infty$
FOREACH sensitized c-input ci driven by FW DO
    IF $V_{I_{DDQ}0,g,ci} < V_{I_{DDQ}0,FW,vec}$ THEN $V_{I_{DDQ}0,FW,vec} = V_{I_{DDQ}0,g,ci}$
ENDFOR

Then, we compute the trapped charge on FW that corresponds to a voltage of $V_{I_{DDQ}0,FW,vec}$. We will explain the details of the computation in Section 4.4. Note that the FW voltage needs to be larger than $V_{I_{DDQ}0,FW,vec}$ for IDDQ detection. Let us call this computed charge $\boldsymbol{Q_{trapped,min,I_{DDQ}0}}$.

Similarly, we compute $\boldsymbol{Q_{trapped,max,I_{DDQ}1}}$ that corresponds to the maximum trapped charge with which vec can detect the open as an IDDQ fault. Note that if a sensitized c-input exists when FW is logic-0, a sensitized c-input must also exist when FW is logic-1, because there will be at least one cell driven by FW, whose other inputs do not have any combinational path to them starting from FW. Then, the IDDQ detection range for vec is ($Q_{trapped,min,I_{DDQ}0}$, $Q_{trapped,max,I_{DDQ}1}$).

Oscillation might occur within this detection range due to possible capacitance from a signal S to FW, such that there is an inverting combinational path from FW to S, and that path is enabled by the given vector. When oscillation occurs, the voltage of FW oscillates with a small amplitude, crossing the logic-0 and logic-1 threshold voltages of a fanout gate back and forth, as demonstrated in [23]. Since the logic threshold voltages will be inside the range defined by the IDDQ threshold voltages, as illustrated in Figure 3, significant amount of current will be drawn from the power supply. At least three CMOS stages are necessary for the inverting path from FW to S for an oscillation to occur, only one stage cannot oscillate. Therefore, there will be at least two more gates other than the fanout gate from FW, whose inputs will be oscillating. These gates will be drawing significant current, too. Therefore, we assume that even if oscillation occurs, it will be

detected as an IDDQ fault.

4.3 Both voltage and current sensing

Due to usage of hardware IDDQ monitors [24, 25], it is now feasible to use hundreds of IDDQ vectors on a tester. Therefore, while a stuck-at vector is applied to a circuit, its IDDQ might also be measured at the same time. In this case, either a voltage discrepancy or a high IDDQ will detect the defect. Again, we will describe the case for stuck-at-0 without loss of generality.

If a vector *vec* detects the stuck-at-0 fault on FW, then we determine the maximum voltage on FW, $\boldsymbol{V_{I_{DDQ}1,FW,vec}}$, such that an IDDQ that is larger than or equal to $I_{DDQ,th}$ still flows through a cell driven by FW. We determine $V_{I_{DDQ}1,FW,vec}$ as the maximum logic-1 IDDQ threshold voltage over all the sensitized c-inputs driven by FW. If the voltage on FW is smaller than or equal to $V_{I_{DDQ}1,FW,vec}$, then the open will be detected by *vec* as either a stuck-at-0 fault or an IDDQ fault. We compute the trapped charge on FW, that corresponds to $V_{I_{DDQ}1,FW,vec}$, as we will describe in Section 4.4. Let us call this computed charge $\boldsymbol{Q_{trapped,max,I_{DDQ}1}}$. Then, the detection range for vector *vec* is $(-\infty, Q_{trapped,max,I_{DDQ}1})$.

Unlike the "stuck-at detection only" case described in Section 4.1, an explicit check for sequential behavior is not needed here because of the following reasons: The first observation we make is that $V_{I_{DDQ}1,FW,vec}$ will be beyond the last drop in the non-linear curve going from left to right on the Q_s-V_{FW} plane. Since a drop in the non-linear curve occurs every time a logic-0 to logic-1 transition happens on a circuit node that has a capacitance to FW, as the voltage of FW increases, and $V_{I_{DDQ}1,FW,vec}$ is larger than the maximum logic-1 threshold of any sensitized c-input driven by FW, there cannot be any drop in the non-linear curve beyond $V_{I_{DDQ}1,FW,vec}$. Let $\boldsymbol{Line_{I_{DDQ}1,FW,vec}}$ denote the vector line crossing the non-linear curve at $V_{I_{DDQ}1,FW,vec}$, as illustrated in Figure 8 for a non-linear curve that makes two drops. Then, any other vector line below $Line_{I_{DDQ}1,FW,vec}$ will intersect the non-linear curve at points where V_{FW} is less than $V_{I_{DDQ}1,FW,vec}$, and these voltage points correspond to either IDDQ detection or stuck-at-0 detection.

4.4 Charge computation

In this subsection we describe how to efficiently compute the total electrical charge on a floating wire FW created by an interconnect open, for a given voltage level V_{FW} on the floating wire and a test vector *vec* applied to the circuit. The computed charge values are used to define the boundaries of the detection ranges for the trapped charge, as described by the preceding subsections.

The total charge on FW has two components: (i) the charge on the transistor gates driven by FW, denoted as $\boldsymbol{Q_{gate}}$, and (ii) the charge on FW due to the wiring capacitances between FW and other nodes (including the substrate and the die surface), denoted as $\boldsymbol{Q_{wire}}$. From the law of charge conservation, the

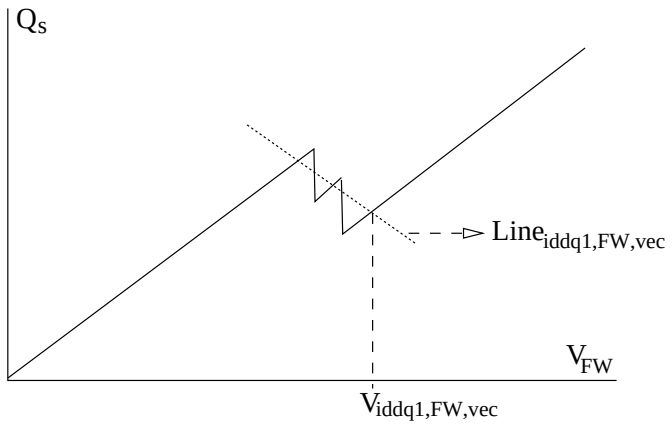

Figure 8: A non-linear curve with two drops due to a fanout of two from FW.

trapped charge on FW is $Q_{trapped} = Q_{gate} + Q_{wire}$.

The equation for Q_{gate} is as follows, where $\boldsymbol{SCI}$ and $\boldsymbol{UCI}$ denote the sets of sensitized and unsensitized c-inputs driven by FW, respectively.

$$Q_{gate} = \sum_{ci \in SCI} (a_{ci} + b_{ci} \cdot V_{FW}) + \sum_{ci \in UCI} Q_{eqn}(ci, V_{FW}, vec) \tag{3}$$

For a sensitized c-input ci, the charge on the transistor gates driven by ci is computed by $(a_{ci} + b_{ci} \cdot V_{FW})$ in Equation 3, where a_{ci} and b_{ci} are the y-intercept and the slope values recorded for ci during our library processing step in Section 3. Recall that two sets of y-intercept and slope values are computed per ci; one for the line passing through the IDDQ threshold and the logic threshold points on the Q-V curve as shown in Figure 3 when the c-input voltage is logic-0, and the other for the line when the c-input voltage is logic-1. This interpolation-based computation in Equation 3 is an efficient way of approximating the charge on a sensitized c-input with reasonable accuracy.

For an unsensitized c-input ci, the charge on the gate of each transistor driven by ci is computed by using Equations 4 and 5, which are taken from Sheu, Hsu, and Ko [15] where the derivations of these equations are explained. Additionally, we included the sensitivity of model parameters to transistor lengths and widths. These equations are for an nMOS transistor. For a pMOS transistor, the right hand sides of Equations 4 and 5 need to be negated together with the V_{gb} and V_{gs} terms.

$$Q_g = \frac{cap \cdot zk1^2}{2} \cdot (-1 + \sqrt{1 + \frac{4 \cdot (V_{gb} - zvfb)}{zk1^2}}) \quad \text{in subthreshold region} \tag{4}$$

$$Q_g = cap \cdot (V_{gs} - zvfb - zphi) \quad \text{with } V_{ds} = 0 \quad \text{in triode region} \tag{5}$$

Any term that starts with "z" in the preceding equations such as $zk1$ or $zvfb$ is a BSIM [18] electrical parameter taking the transistor size into account, and we compute it as follows [26]:

$$zP = P + \frac{P_L}{L - DL} + \frac{P_W}{W - DW}$$

where P is a process parameter such as vfb or phi, P_L and P_W are the length and width sensitivities of parameter P, W and L are the drawn transistor width and length, and DW and DL are the size changes to W and L due to various fabrication steps. The values of P, P_L, P_W, DL, and DW are all determined by the fabrication process. We obtained the values of all the BSIM parameters from MOSIS.

Finally, $cap = C_{ox} \cdot (W - DW) \cdot (L - DL)$ where C_{ox} is the gate-oxide capacitance per unit area.

The saturation region is not included in Equations 4 and 5, because, by definition, no current will be flowing through the transistors driven by an unsensitized c-input. For the same reason, V_{ds} is zero in Equation 5. These gate charges are added up to find $Q_{eqn}(ci, V_{FW}, vec)$ in Equation 3, where vec stands for the test vector applied to the circuit. We could not use interpolation to compute $Q_{eqn}(ci, V_{FW}, vec)$; because, unlike a sensitized c-input, V_{FW} alone is not sufficient to determine the drain/source voltages of the transistors driven by an unsensitized c-input.

Equation 6 shows the expression for Q_{wire}. $\boldsymbol{C0}$ and $\boldsymbol{C1}$ denote the sets of capacitances from FW to other nodes that are at logic-0 and at logic-1, respectively. Recall from Section 2.1 that our algorithm uses a range for each wiring capacitance due to process variations and possible accuracy limitations of capacitance extraction tools. Equation 7 shows how to pick the worst case values for C_{w0} and C_{w1} in order to guarantee detection. The detection of the open might occur below V_{FW}, for instance, when V_{FW} is the maximum logic-0 voltage for the floating wire, and the vector applied is a test for FW stuck-at-0. In this case, $C_{w1,max}$ will be used for C_{w1} as the worst case, because the corresponding wire $w1$ is at logic-1, thus, pulling the floating wire voltage up towards logic-1 (non-detection of the open) with a strength proportional to the size of C_{w1}.

$$\begin{aligned} Q_{wire} \quad = \quad & \sum_{C_{w0} \in C0} C_{w0} \cdot V_{FW} + \sum_{C_{w1} \in C1} C_{w1} \cdot (V_{FW} - V_{DD}) + \\ & C_{FW\_surf} \cdot (V_{FW} - V_{surf}) \end{aligned} \tag{6}$$

$$C_{w0}(C_{w1}) = \begin{cases} C_{w0,min}(C_{w1,max}) & \text{if } FW \text{ voltage needs to be} < V_{FW} \text{ for detection} \\ C_{w0,max}(C_{w1,min}) & \text{if } FW \text{ voltage needs to be} > V_{FW} \text{ for detection} \end{cases} \tag{7}$$

If the die surface does not have a large enough resistivity, then the capacitance between FW and the surface needs to be considered, also. Given a voltage range $V_{surf,min}$ and $V_{surf,max}$ the die surface can

acquire, the last term in Equation 6 is used for modeling the worst case effect of the die surface. Again, either $C_{FW\_surf,min}$ or $C_{FW\_surf,max}$ is used for C_{FW_surf}, and either $V_{surf,min}$ or $V_{surf,max}$ is used for V_{surf} depending on whether V_{FW} is larger than V_{surf} or not.

Transistor drain/source voltages need to be determined to decide whether a transistor driven by an unsensitized c-input is in subthreshold or triode region to compute its gate charge using Equation 4 or 5. Also, a floating wire will sometimes go over cells in a layout; thus, it will have capacitances to internal nodes of these cells, such as node $n1$ in Figure 5. These internal nodes are also connected to transistor drain/source terminals, whose voltages are used in Equation 6 instead of GND and V_{DD} voltages used for signal wires. We use four voltage levels for an internal node, which are GND, V_{DD}, max_n, and min_p, where **max_n** is the maximum voltage an internal node in an n-network can achieve through a path to V_{DD}, and **min_p** is the minimum voltage an internal node in a p-network can achieve through a path to GND, as also used by [2] for network breaks.

The voltage of an internal node is determined as done in [2], details of which are not repeated here to save space. Briefly, sum-of-products Boolean expressions are used to specify the cell input conditions for each internal node to have a path to V_{DD} or GND. These sum-of-products expressions are computed during our library processing step for each internal node in the library, but they are evaluated for every new vector during the fault simulation of an interconnect open.

5 Experiments and Results

We used the two metal layer channel routed layouts of the ISCAS85 circuits for our experiments. Since we consider each via (the contact between metal 1 and 2) as an interconnect open site, we process each layout with a program that is an extended version of *Carafe* [27] in order to analyze each via to find out whether a floating wire is created if the via is broken. There are some redundant vias in the layouts, which do not create an open when broken. The program removes each via that can create a floating wire from the layout, and labels the wire pieces in a systematic fashion. Then, we run *Magic* using the HP 0.6μ parameters in the technology file 8.2.8 from MOSIS to extract the capacitances from the layout.

We would like to emphasize that the main purpose of our results in this section is to demonstrate the functionality of our simulator, rather than coming up with conclusions that can be generalized to all VLSI circuits.

There is a metal-2 wire across the height of each MCNC cell used in the ISCAS85 layouts for every input port of the cell. A via connects this metal-2 wire to a small piece of metal-1, which is connected to poly that drives transistor gates. When this via is broken, the floating poly together with a small piece of metal-1 form a very short floating wire (s-FW), which have very small capacitances to neighboring nodes and to the

die surface. The numbers of breaks corresponding to these vias are listed in the second column of Table 1. The remaining via-breaks are considered to create "long" floating wires (*l-FW*), and are listed in the third column. The IDDQ and stuck-at vectors are generated by *Nemesis* [28]. The IDDQ coverages in the fifth column are based on the pseudo-stuck-at fault model.

Ct.	# of *s-FW* via-breaks	# of *l-FW* via-breaks	# of IDDQ vectors	IDDQ cov.	# of stuck-at vectors	stuck-at cov.
c432	322	717	30	98.78 %	55	97.16 %
c499	386	856	52	100.00 %	53	98.98 %
c880	510	1144	36	100.00 %	63	100.00 %
c1355	884	1528	81	100.00 %	85	99.57 %
c1908	720	1551	77	99.73 %	99	99.28 %
c2670	1286	3141	42	97.78 %	117	96.33 %
c3540	1854	4420	67	97.71 %	175	96.58 %
c5315	2938	7652	60	99.62 %	137	98.79 %
c6288	4232	8260	34	99.40 %	33	99.42 %
c7552	3464	9054	86	99.62 %	221	98.89 %

Table 1: Via-break and test vector statistics

For a given manufactured die, the wiring capacitances are fixed, even though their values may not be accurately known due to limitations of extraction tools and process variations. So, we decided to use the *Magic* extracted capacitance values as the real values, and assume that it is theoretically possible to modify the given circuit layout to exactly match these capacitance values, and the actual design is this modified layout. We also assumed that the die surface effect of Section 2.4 does not apply. Table 2 shows our simulation results. Columns 2 to 5 show the % of *s-FW* and *l-FW* opens guaranteed to be detected no matter what the actual trapped charge is[1], when stuck-at (SA) vectors are used alone or in combination with IDDQ vectors. In columns 2 through 5, if an open is declared as detected, it means that the open will be detected by the test set independent of the amount and the polarity of the trapped charge on the floating wire created by that open.

SA vectors are good at detecting interconnect opens when the magnitude of the actual trapped charge is large, forcing the floating wire to behave as stuck-at-0 or stuck-at-1. In contrast, IDDQ vectors tend to detect opens when the trapped charge magnitude is small, resulting in floating wire voltages being pulled to

[1]Note that excessive trapped charge may create a dangerous voltage level such that a gate-oxide punch through may occur, creating a gate-oxide short coupled with an interconnect open. We do not consider this case and leave it to future research.

the vicinity of $V_{DD}/2$ by wiring and transistor capacitances when the chip is powered up. This is why SA and IDDQ vectors together produce much better guaranteed coverages in columns 4 and 5, where detection is guaranteed no matter what the actual trapped charge is.

In all our experiments, the two IDDQ threshold voltages for each standard cell c-input were computed as described in Section 3, which correspond to $50\mu A$ of IDDQ current flowing through the standard cell being processed. If $I_{DDQ,th} = 50\mu A$ additional current in either the single threshold or the current signatures [21] IDDQ test technique, which is described in Section 3, does not provide enough resolution to differentiate a defective chip, then $I_{DDQ,th}$ needs to be increased. Note that this will make the IDDQ detection range defined by the two IDDQ threshold voltages for each c-input narrower, which will in turn reduce the effectiveness of IDDQ vectors.

If the initial voltage on a floating wire due to the trapped charge can be bounded, then the detection ranges our algorithm computes, as explained in Section 4, can be compared to the size of the total possible range to give a probabilistic range coverage number. For example; if the trapped charge voltage can be guaranteed to fall in the [-1.0V, 1.0V] range for any interconnect open, and the detection ranges computed by our fault simulator are ($-\infty$, -0.3V) and (0.25V, ∞) for a particular open, then the trapped charge range coverage for that open will be

$$[(-0.3V - (-1.0V)) + (1.0V - 0.25V)] \div (1.0V - (-1.0V)) = 0.725 = 72.5\%$$

assuming a homogeneous distribution of actual trapped charge voltages in the [-1.0V, 1.0V] range. We define the **range coverage for a chip** as the average value of range coverages over all the interconnect opens. This is what we did for the rest of Table 2. Konuk and Ferguson [12] reported that trapped charge on floating metal wires connected to transistor gates can create voltages in the range between -1V to 1V, but 75% of the measurements were between -0.5V and 0.5V for a set of experimental chips fabricated with an HP 0.8μ process. Therefore, assuming that the actual trapped charge voltages are bounded by [-1.0V, 1.0V], we obtained the range coverage numbers in columns 6 and 7. Using the [-0.5V, 0.5V] range decreased the coverage numbers, as we expected, because we used only the SA vectors, and SA vectors get worse in detection when the trapped charge magnitude gets smaller around 0V. Still, these range coverage numbers are much better than the guaranteed coverage numbers in columns 2 and 3.

In order to be certain of the coverage of a set of test vectors, one needs to take into account the variation in the capacitance values due to extraction tool inaccuracies and process variations, plus the effect of the die surface. Assuming that the trapped charge voltage is bounded by [-0.5V, 0.5V], and using either V_{DD} or GND for the die surface voltage depending on which one being the worst case, we obtained the range coverage percentages in Table 3. V_{DD} is 3.3V for all our experiments. This table shows the sensitivity of coverage to capacitance variation and to different sets of tests. +/-30% variation in capacitance means that

Circuit	% opens guaranteed to be detected with no bound on the trapped charge				Trapped charge range cov. (%)			
	SA only		SA and IDDQ		SA only, -1.0V to 1.0V trapped Q volt.		SA only, -0.5V to 0.5V trapped Q volt.	
	s-FW	*l-FW*	*s-FW*	*l-FW*	*s-FW*	*l-FW*	*s-FW*	*l-FW*
c432	0.00	4.46	95.03	98.19	87.21	87.44	79.65	76.93
c499	0.00	3.04	97.93	98.83	88.53	86.98	79.50	74.96
c880	0.00	17.92	100.00	99.83	89.43	92.61	80.12	86.12
c1355	0.00	10.14	99.10	99.80	88.74	91.50	78.49	84.53
c1908	0.00	6.77	98.75	99.48	89.29	89.97	80.96	81.06
c2670	0.00	5.19	93.86	97.33	86.52	89.89	77.46	83.61
c3540	0.00	16.99	93.91	97.22	86.85	91.04	79.26	85.27
c5315	0.00	11.43	98.03	97.28	88.78	92.06	80.06	86.76
c6288	0.00	18.67	99.13	99.84	88.35	91.51	77.71	84.44
c7552	0.00	6.57	97.89	99.14	88.89	90.80	80.32	85.57

Table 2: Via-break coverages assuming **no surface effect** and using **extracted capacitance values**.

the actual value for a wiring capacitance is within the range from $0.7 * C_{extracted}$ to $1.3 * C_{extracted}$.

Note that the coverage numbers in column 3 for *l-FW* opens is particularly low. The reason is the effect of the die surface, because the same column in Table 4 has coverages that are about three times as large. The only difference in the preparation of Table 4 is that V_{surf} is assumed to be bounded by [$0.25V_{DD}$, $0.75V_{DD}$] rather than [0V, V_{DD}]. The observation that the die surface has an extremely significant effect on the coverage numbers for long wires is not surprising, because long floating wires are affected by the die surface more than the short floating wires are. However, this result should not be generalized, because the ISCAS85 layouts use only 2 metal layers, whereas modern designs have up to 6 layers of metal, which can easily hide floating wires in layers 1 and 2 from the die surface.

Comparing column 3 to column 5 in Table 3 shows that decreasing the capacitance variation from 30% to 15% did not have a large effect on the coverage numbers, at all.

Using IDDQ vectors in addition to SA vectors boosts the coverage numbers as listed in columns 6 through 9. The *l-FW* opens get the largest boost in coverage from adding the IDDQ vectors, which compensate the adverse effect of the die surface on the *l-FW* opens. Decreasing the capacitance variation from 30% to 15% helps coverage about 12 percentage points as shown in columns 7 and 9. Again, this is a smaller effect compared to the effect of adding IDDQ vectors or reducing the voltage variation of the die surface. The *s-FW* opens achieve almost full coverage even with 30% capacitance variation as shown in column 6.

Circuit	Range coverage %'s for trapped charge, with V_{surf} between V_{DD} and GND					
	Capacitance variation (SA tests only)			Capacitance variation (SA and IDDQ)		
	+/-30%		+/-15%	+/-30%		+/-15%	
	s-FW	*l-FW*	*s-FW*	*l-FW*	*s-FW*	*l-FW*	*s-FW*	*l-FW*
c432	67.21	20.30	71.32	27.23	100.00	80.76	100.00	91.44
c499	66.42	20.34	71.02	27.62	99.56	75.88	99.81	87.30
c880	66.67	23.62	71.08	32.11	100.00	81.66	100.00	90.85
c1355	64.84	23.54	69.29	31.50	100.00	78.55	100.00	90.06
c1908	67.84	19.57	72.31	27.50	99.95	72.85	99.98	85.98
c2670	64.21	17.74	68.57	25.59	99.63	66.28	99.65	79.64
c3540	67.06	19.47	71.10	27.66	99.63	70.48	99.67	84.12
c5315	67.08	18.99	71.37	27.58	99.97	64.89	99.97	78.20
c6288	64.02	22.24	68.47	30.49	99.62	75.85	99.62	87.31
c7552	66.97	17.35	71.44	25.51	99.99	61.57	100.00	76.43

Table 3: Range coverages for trapped charge with surface voltage varying betweenV_{DD} and GND

Table 4 shows that the effect of the die surface voltage is pretty significant on the coverage numbers. Again, we would like to emphasize that ISCAS85 layouts have only 2 metal layers, and this is an expected result for them. Designs with more metal layers will not be as sensitive to die surface voltage variation.

As described earlier, if the magnitude of the trapped charge range gets larger, then the coverage of this range by stuck-at vectors is expected to get better. We ran our simulator using the parameters for column 6 and 7 in Table 3, but using the [-1.0V, 1.0V] range for the trapped charge instead of the [-0.5V, 0.5V] range. The range coverage numbers for the *l-FW* opens increased between 4 and 9 percentage points, as expected. The range coverage numbers for the *s-FW* opens stayed almost the same, which were almost full coverage, anyway.

If the trapped charge bounds stay the same while the V_{DD} voltage decreases, then the net effect will be as if the V_{DD} voltage stayed the same and the trapped charge bounds increased, which actually helps the trapped charge range coverage, as discussed above. Here we assume that the logic and IDDQ threshold voltages of the standard cells scale down at the same rate as the V_{DD} voltage does. However, if the ratio of the distance between the logic-0 and logic-1 IDDQ threshold voltages to the V_{DD} value decreases with decreasing V_{DD}, then this will have a diminishing effect on the range coverage of IDDQ vectors. Further experiments can be performed using our simulator using a process technology and a cell library that are designed for lower V_{DD} to find out how the coverage of IDDQ vectors will be affected.

In case IDDQ vectors cannot be applied at high speed, the number of IDDQ vectors might be limited.

Circuit	Range coverage %'s for trapped charge, with V_{surf} between $0.75V_{DD}$ and $0.25V_{DD}$									
	Capacitance variation (SA tests only)				Capacitance variation (SA and IDDQ)					
	+/-30%		+/-15%		+/-30%		+/-15%			
	s-FW	*l-FW*	*s-FW*	*l-FW*	*s-FW*	*l-FW*	*s-FW*	*l-FW*		
c432	72.13	60.37	75.82	68.70	100.00	100.00	100.00	100.00		
c499	70.91	58.70	75.13	67.28	99.76	99.72	100.00	99.94		
c880	72.15	69.09	76.10	77.71	100.00	100.00	100.00	100.00		
c1355	70.53	67.52	74.48	76.29	100.00	100.00	100.00	100.00		
c1908	72.74	63.64	76.80	72.52	99.97	99.91	100.00	99.99		
c2670	69.65	63.59	73.54	72.94	99.65	99.76	99.66	99.83		
c3540	71.89	66.48	75.51	75.82	99.66	99.73	99.69	99.81		
c5315	72.33	67.04	76.17	76.63	99.97	99.93	99.97	99.99		
c6288	69.66	67.16	73.63	75.91	99.62	99.85	99.62	99.85		
c7552	72.26	64.72	76.25	74.30	100.00	99.95	100.00	100.00		

Table 4: Range coverages for trapped charge with surface voltage varying between 0.825V and 2.475V

In order to find the impact of this at least on the ISCAS85 circuits, we used only 10% of the IDDQ vectors available for each circuit. For simplicity, we just picked the first 10% from each IDDQ test set, rather than picking the 10% that will give the largest pseudo-stuck-at coverage. We used the stuck-at test sets unmodified. We obtained the results in Table 5 with V_{surf} bounded by [$0.25V_{DD}$, $0.75V_{DD}$] and capacitance variation +/-30%. Interestingly, the range coverage numbers are pretty close to the ones in columns 6 and 7 of Table 4, where full IDDQ and stuck-at tests were used. The second row in Table 5 shows the pseudo-stuck-at fault coverages of the IDDQ vectors used. They are pretty low compared to the ones from full IDDQ tests shown in the "IDDQ cov." column of Table 1. Still, the range coverages are pretty high in Table 5. There are two factors contributing to achieving high range coverage with IDDQ vectors that have low pseudo-stuck-at coverage.

First, an IDDQ vector that covers the pseudo-stuck-at-0 fault at input a of a basic cell, as defined in Section 3, will cause high IDDQ current inside that cell if input a acquires a voltage between the IDDQ threshold voltages of a due to an interconnect open. An IDDQ vector that covers the pseudo-stuck-at-1 fault at a will be able to do the same. Therefore, one IDDQ vector that covers either the pseudo-stuck-at-0 or pseudo-stuck-at-1 can be sufficient; covering both faults will not be necessary. Second, if a floating wire created by an interconnect open has a fanout of more than one, then an IDDQ vector that covers a pseudo-stuck-at fault at any of the fanout branches can create high IDDQ, and thus causes that open to be detected, even though none of the IDDQ vectors used cover a pseudo-stuck-at fault at other branches. Therefore, even

a small set of IDDQ vectors can be very useful in achieving high coverage for interconnect-opens.

	c432	c499	c880	c1355	c1908	c2670	c3540	c5315	c6288	c7552
IDDQ cov.	55.78	73.68	67.82	78.88	70.40	54.67	69.51	71.10	65.96	71.84
s-FW	93.99	95.20	97.19	97.71	95.07	90.98	96.08	96.77	96.30	96.13
l-FW	94.02	96.76	98.09	99.49	96.88	92.50	97.26	97.96	98.25	96.56

Table 5: Range coverage percentages with stuck-at vectors and using only the first 10% of the IDDQ vectors with V_{surf} between 0.825V and 2.475V, and capacitance variation +/-30%

We used an HP-735 99MHz workstation with 180MB memory for our fault simulations. The maximum wall clock time in our experiments was 4 minutes 24 seconds for circuit c7552, which shows us promise for the feasibility of a fault simulator for real world chips.

6 Conclusion

We presented a highly accurate and efficient fault simulator for interconnect opens, which take almost all factors that can affect the voltage of an open into account. Our results from ISCAS85 circuit layouts show that combination of high coverage stuck-at and IDDQ test sets can result in high coverage of interconnect opens, especially when the die surface voltage is bounded, while stuck-at tests alone may not always guarantee such high coverage.

References

[1] C.F. Hawkins, J.M. Soden, A.W. Righter, and F.J. Ferguson. Defect classes - an overdue paradigm for CMOS IC testing. In *Proceedings of ITC*, Oct. 1994.

[2] H. Konuk, F.J. Ferguson, and T. Larrabee. Charge-based fault simulation for CMOS network breaks. *IEEE Transactions on Computer-Aided Design*, pages 1555–67, Dec. 1996 (http://sctest.cse.ucsc.edu/papers/publications.html).

[3] C. Di and J.A.G. Jess. On accurate modeling and efficient simulation of CMOS opens. In *Proceedings of International Test Conference*, pages 875–882, October 1993.

[4] M. Favalli, M. Dalpasso, P. Olivo, and B. Ricco. Modeling of broken connections faults in CMOS ICs. In *Proceedings of European Design and Test Conference*, 1994.

[5] W.M. Maly, P.K. Nag, and P. Nigh. Testing oriented analysis of CMOS ICs with opens. In *Proceedings of International Conference on Computer-Aided Design*, Nov. 1988.

[6] V.H. Champac, A. Rubio, and J. Figueras. Electrical model of the floating gate defect in CMOS IC's: Implications on IDDQ testing. *IEEE Transactions on Computer-Aided Design*, March 1994.

[7] M. Renovell and G. Cambon. Electrical analysis and modeling of floating-gate fault. *IEEE Transactions on Computer-Aided Design*, pages 1450–1458, November 1992.

[8] H. Xue, C. Di, and J.A.G. Jess. Probability analysis for CMOS floating gate faults. In *Proceedings of European Design and Test Conference*, 1994.

[9] D.B.I. Feltham and W. Maly. Physically realistic fault models for analog CMOS neural networks. *IEEE Journal of Solid-State Circuits*, Sep. 1991.

[10] K.M. Thompson. Intel and the myths of test. In *The Keynote Address of International Test Conference*, October 1995.

[11] H. Konuk. Fault simulation of interconnect opens in digital CMOS circuits. In *Proceedings of International Conference on Computer-Aided Design*, November 1997.

[12] H. Konuk and F.J. Ferguson. An unexpected factor in testing for CMOS opens: The die surface. In *IEEE VLSI Test Symposium*, 1996 (http://sctest.cse.ucsc.edu/papers/publications.html).

[13] H. Konuk. Testing for opens in digital CMOS circuits. *Ph.D. Thesis, Computer Eng. Dept., UC Santa Cruz*, Dec. 96 (http://sctest.cse.ucsc.edu/papers/publications.html).

[14] A.J. van Genderen and N.P. van der Meijs. *Space3d Capacitance Extraction User's Manual.* Delft University of Technology, 1997 (http://cas.et.tudelft.nl/ space).

[15] B.J. Sheu, W.-J. Hsu, and P.K. Ko. An MOS transistor charge model for VLSI design. *IEEE Transactions on Computer-Aided Design*, pages 520–527, April 1988.

[16] S. Johnson. Residual charge on the faulty floating gate MOS transistors. In *Proceedings of International Test Conference*, October 1994.

[17] MCNC. *http://www.mcnc.org.*

[18] Meta-Software. *HSPICE User's Manual: Elements and Models.* 1992.

[19] MOSIS. *http://www.mosis.org.*

[20] A.D. Singh, H. Rasheed, and W.W. Weber. IDDQ testing of CMOS opens: An experimental study. In *Proceedings of International Test Conference*, 1995.

[21] A.E. Gattiker and W. Maly. Current signatures: Aplication. In *Proceedings of International Test Conference*, November 1997.

[22] J.A. Waicukauski, E.B. Eichelberger, D.O. Forlenza, E. Lindbloom, and Th. McCarthy. Fault simulation for structured VLSI. In *VLSI Systems Design*, pages 20–32, Dec. 1985.

[23] H. Konuk and F.J. Ferguson. Oscillation and sequential behavior caused by opens in the routing in digital CMOS circuits. *IEEE Transactions on Computer-Aided Design*, pages 1200–10, Nov. 1998 (http://alumni.cse.ucsc.edu/ haluk/publications.html).

[24] P.C. Maxwell, R.C. Aitken, K.R. Kollitz, and A.C. Brown. IDDQ and AC scan: The war against unmodelled defects. In *Proceedings of International Test Conference*, 1996.

[25] H.A.R. Manhaeve, P.L. Wrighton, J. van Sas, and U. Swerts. An off-chip IDDQ current measurement unit for telecommunication asics. In *Proceedings of ITC*, 1994.

[26] G. Massobrio and P. Antognetti. *Semiconductor Device Modeling with SPICE*. McGraw-Hill, 1993.

[27] A. Jee and F.J. Ferguson. Carafe: An inductive fault analysis tool for CMOS VLSI circuits. In *Proceedings of the IEEE VLSI Test Symposium*, 1993.

[28] T. Larrabee. Test pattern generation using Boolean satisfiability. *IEEE Transactions on Computer-Aided Design*, pages 6–22, January 1992.